
# Feature Engineering

# Ingeniería de Features

*“At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used.”*

*— Prof. Pedro Domingos*

¿Por qué ingeniería de features?

Conversión de variables

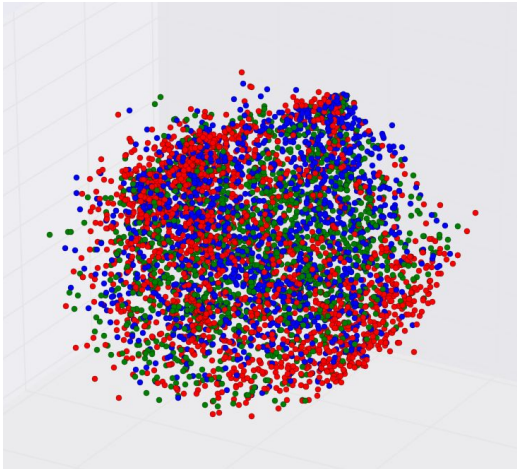
Missings

Selección de variables

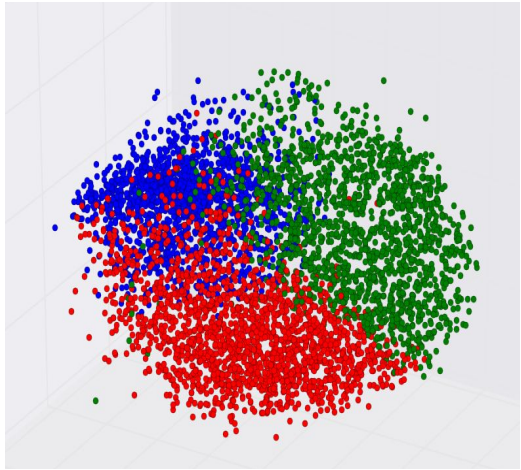
Reducción dimensional

# Garbage in... garbage out

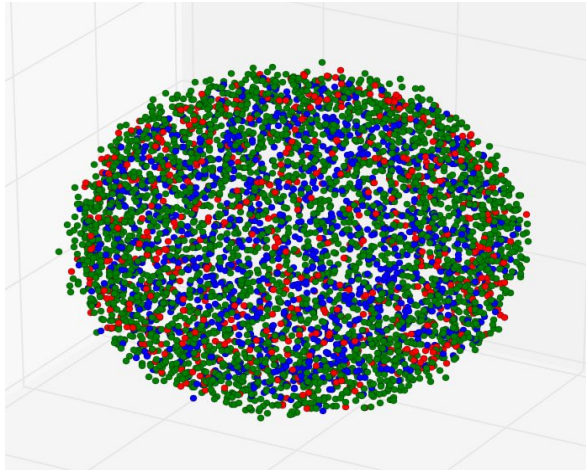

Clasificar vinos según distintos atributos: ¿malbec? ¿cabernet? ¿merlot?



<Precio, Color,  Origen>



<Acidez, Densidad, Color>



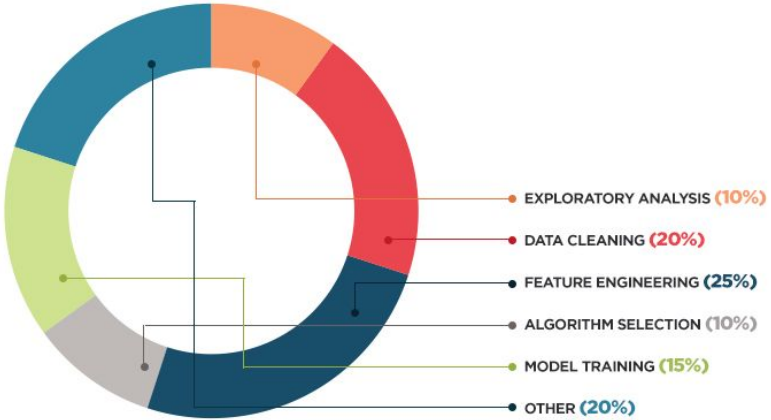
<Material Corcho, Litros Botella, Origen>

# ¿Por qué trabajar sobre los features?

**Atributo (feature)**

Un pedazo de información potencialmente bueno para obtener predicciones útiles.

*“Good features allow a simple model to beat a complex model” (Peter Norvig)*



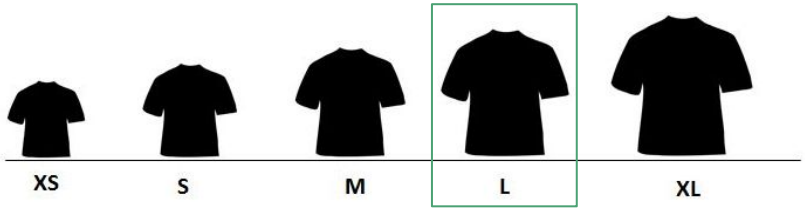

**Ingeniería de Features**

Tareas relacionadas a diseñar y manipular un conjunto de features para tareas de AA.

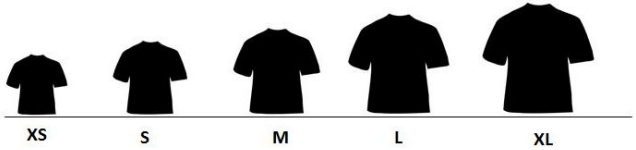
**Tareas**

- Extracción.
- Conversión de features.
- Tratamiento de datos faltantes.
- Selección de variables.
- Combinación de variables.
- Reducción dimensional.
- etc.

# Conversión de variables




$x_2 = 4$



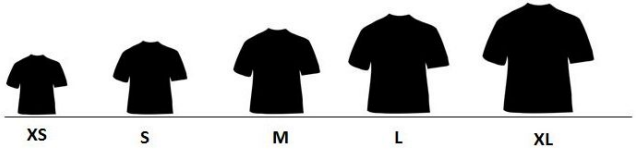

# Conversión de variables

Los modelos sólo entienden vectores de números. ¿Si las features no son números?

**Tipos de variables a tratar**

- Numéricas: *edad.*
- Categóricas: *nacionalidad.*
- Ordinales: *tamaño remera.*
- Fechas / Hora del día
- Ubicaciones (lat/long)

```
<3.2, 1, 0, 2.4, 6.6, 4.6, 5.3, 14.3>
```




# Conversión de variables

Los modelos sólo entienden vectores de números. ¿Si las features no son números?

**Tipos de variables a tratar**

- Numéricas: *edad*.
- Categóricas: *nacionalidad*.
- Ordinales: *tamaño remera*.
- Fechas / Hora del día
- Ubicaciones (lat/long)

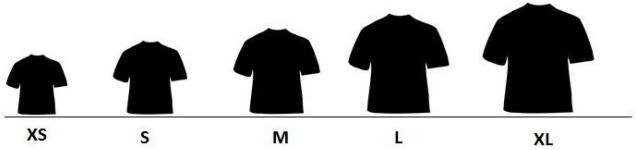
<3.2, 1, 0, 2.4, 6.6, 4.6, 5.3, 14.3>

¿Cómo convertir Categóricas? (Parte I)

- ¿Asignar números? ordinales: OK
  categóricas: en general, mala idea (¿rojo > verde?)

- **One-Hot encoding**
  Por cada variable categórica, crear variables binarias. Una por cada posible categoría.
  **Problema:** “dummy trap” (problema de la colinealidad)

- **Dummy Variables**
  Por todas las categorías menos una, crear variables dummy (binarias). La ausencia de todas significa presencia de la categoría no encodeada.




# Conversión de variables

Los modelos sólo entienden vectores de números. ¿Si las features no son números?

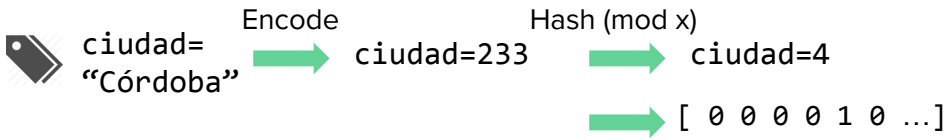
**Tipos de variables a tratar**

- Numéricas: *edad*.
- Categóricas: *nacionalidad*.
- Ordinales: *tamaño remera*.
- Fechas / Hora del día
- Ubicaciones (lat/long)

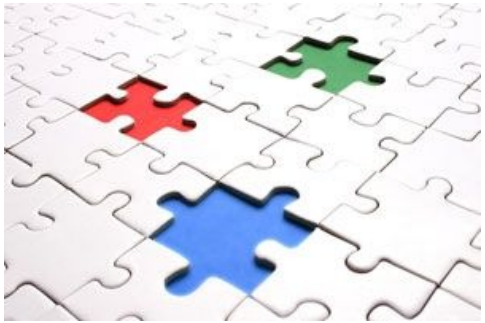
<3.2, 1, 0, 2.4, 6.6, 4.6, 5.3, 14.3>

¿Cómo convertir Categóricas? (Parte II)

- Ejemplo variable: *dirección IP*.
- Dummy variables... ¿o no?
  - Espacio
  - Tiempo de entrenamiento
  - Memoria
  - Problemas de dimensionalidad: ¡Overfitting!
  - ¿Qué sucede con categorías nuevas? (no vistas en training)
- Otras opciones:
  - Bin-counting Scheme
  - **Hashing Trick**



# Missing Values



# Manejo de “Missings”

¿Qué hacer si faltan datos en nuestra base de datos?

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

“Obviously, the best way to treat missing data is not to have them”.

**¿Por qué pueden faltar?**

**MCAR** (missing completely at random)
P(missing) para todas las instancias es la misma y no depende de las medidas de otras variables.
Ej: Se perdió la respuesta para una encuesta

**MAR** (missing at random)
P(missing) depende de la información observada.
Ej: missing en nota del recuperatorio (depende de valor observado para nota del primer parcial)

**MNAR** (missing not at random)
P(missing) está relacionada con los valores perdidos.
Ej: ¿cuánto gana? (si es muy alto quizás no contestan)

# Manejo de “Missings”

¿Qué hacer si faltan datos en nuestra base de datos?

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

“Obviously, the best way to treat missing data is not to have them”.

**¿Por qué pueden faltar?**

**MCAR** (missing completely at random)
P(missing) para todas las instancias es la misma y no depende de las medidas de otras variables.
Ej: Se perdió la respuesta para una encuesta

**MAR** (missing at random)
P(missing) depende de la información observada.
Ej: missing en nota del recuperatorio (depende de valor observado para nota del primer parcial)

**MNAR** (missing not at random)
P(missing) está relacionada con los valores perdidos.
Ej: ¿cuánto gana? (si es muy alto quizás no contestan)

# Manejo de “Missings”

¿Qué hacer si faltan datos en nuestra base de datos?

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

“Obviously, the best way to treat missing data is not to have them”.

**¿Por qué pueden faltar?**

**MCAR** (missing completely at random)
P(missing) para todas las instancias es la misma y no depende de las medidas de otras variables.
Ej: Se perdió la respuesta para una encuesta

**MAR** (missing at random)
P(missing) depende de la información observada.
Ej: missing en nota del recuperatorio (depende de valor observado para nota del primer parcial)

**MNAR** (missing not at random)
P(missing) está relacionada con los valores perdidos.
Ej: ¿cuánto gana? (si es muy alto quizás no contestan)

# Manejo de “Missings”

¿Qué hacer si faltan datos en nuestra base de datos?

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

“Obviously, the best way to treat missing data is not to have them”.

**¿Por qué pueden faltar?**

**MCAR** (missing completely at random)
P(missing) para todas las instancias es la misma y no depende de las medidas de otras variables.
Ej: Se perdió la respuesta para una encuesta

**MAR** (missing at random)
P(missing) depende de la información observada.
Ej: missing en nota del recuperatorio (depende de valor observado para nota del primer parcial)

**MNAR** (missing not at random)
P(missing) está relacionada con los valores perdidos.
Ej: ¿cuánto gana? (si es muy alto quizás no contestan)

# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)

XGBoost

# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

# Manejo de “Missings”

**Posibles soluciones**

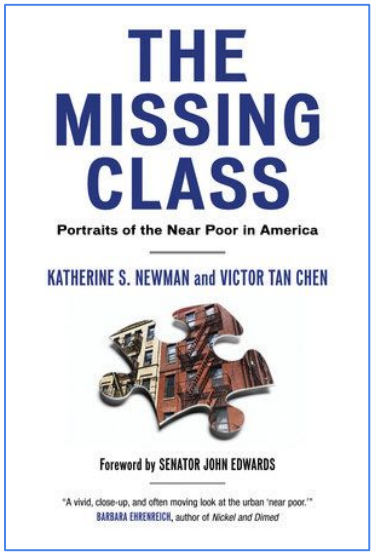
- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)



# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | 41.0 | Programadora | Alto | 2.0 | False |
| Hernán Wilkinson | 41.0 | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | 78.0 | Conductora | Regular | -10.0 | True |

# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)

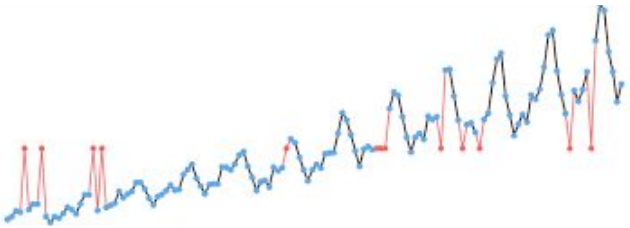
| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | NaN | Programadora | None | NaN | False |
| Hernán Wilkinson | NaN | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | NaN | Conductora | Regular | -10.0 | True |

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? |
|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True |
| Ada lovelace | 41.0 | Programadora | Alto | 2.0 | False |
| Hernán Wilkinson | 41.0 | Smalltalkero | Regular | 7.0 | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True |
| Mirta Legrand | 78.0 | Conductora | Regular | -10.0 | True |

# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Columnas indicadoras (dummy variables)



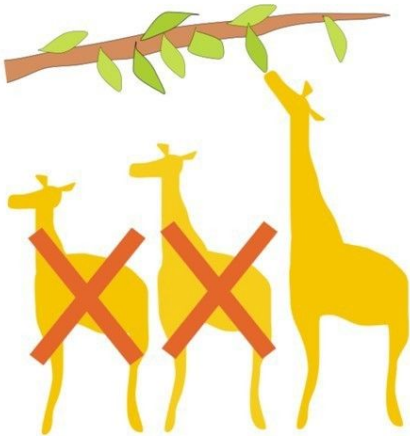
# Manejo de “Missings”

**Posibles soluciones**

- Utilizar algoritmos que puedan trabajar con datos missing (no sklearn)
- Eliminar datos con problemas
  - ¿Eliminar filas o columnas?
  - ¿Está bien perder instancias de entrenamiento?
  - ¿Y en test cómo hago?
- Convertir NaNs en una categoría más (para categóricas)
- Rellenar (Imputers)
  - Generales:
    - Media / Mediana / Moda / Constante
    - Random Forest Imputer, KNN imputer, MICE, etc
  - Para series temporales:
    - Last Observation Carried Forward (LOCF)
    - Next Observation Carried Backward (NOCB)
    - Interpolación (lineal, pesada, splines, etc)
- Imputación + columnas indicadoras (dummy variables)

| Nombre | Edad | Profesión | Estatura | Skills en Python | ¿Vive? | edad_was_nan |
|---|---|---|---|---|---|---|
| Emanuel Ginobili | 41.0 | Basquetbolista | Alto | 2.0 | True | False |
| Ada lovelace | 41.0 | Programadora | Alto | 2.0 | False | True |
| Hernán Wilkinson | 41.0 | Smalltalkero | Regular | 7.0 | True | True |
| Chuck Norris | 78.0 | Todas | Regular | 10.0 | True | False |
| Mirta Legrand | 78.0 | Conductora | Regular | -10.0 | True | True |

# Selección de variables



- Reduce la dimensión
- Favorece a la generalización
- Acelera la velocidad
- Mejora la interpretabilidad

# Selección de variables

Idea: Reducir la dimensión mediante eliminación de variables poco útiles

**Filter**
Ranking generado a través de algún método estadístico.

**Wrapper**
Considera el problema de selección de features como un problema de búsqueda.

**Embedded**
Rankear variables según métodos internos de cada algoritmo

# Selección de variables

Idea: Reducir la dimensión mediante eliminación de variables poco útiles

**Filter**
Ranking generado a través de algún método estadístico.

**Wrapper**
Considera el problema de selección de features como un problema de búsqueda.

**Embedded**
Rankear variables según métodos internos de cada algoritmo

**Filter** (“single factor analysis”)
Objetivo: **¿Qué variable afecta más valor a predecir?**

Test univariados (suponen independencia condicional):

- T-test / Anova (para datos continuos).
- Chi-cuadrado, Information Gain  (para datos categóricos).
- Pearson's correlation con Y.
- Gini Index.
- Un modelo por variable
- etc

# Selección de variables

Idea: Reducir la dimensión mediante eliminación de variables poco útiles

**Filter**
Ranking generado a través de algún método estadístico.

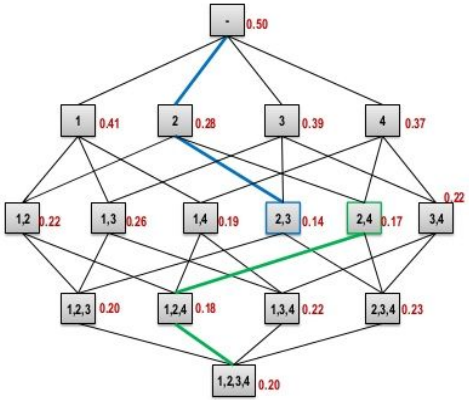
**Wrapper**
Considera el problema de selección de features como un problema de búsqueda.

**Embedded**
Rankear variables según métodos internos de cada algoritmo

**Wrapper**
Objetivo: **¿Cuál es la mejor combinación de variables?**

Se preparan combinaciones, se evalúan y se comparan a través de entrenar un modelo por combinación y luego, medir su performance.

- Heurísticas Greedy:
  - Forward selection
  - Backward selection
  - Random selection
- Best first search
- Random Climbing



# Selección de variables

Idea: Reducir la dimensión mediante eliminación de variables poco útiles

**Filter**
Ranking generado a través de algún método estadístico.

**Wrapper**
Considera el problema de selección de features como un problema de búsqueda.

**Embedded**
Rankear variables según métodos internos de cada algoritmo

**Embedded**
Objetivo: **¿Cuál es la importancia para predecir?**

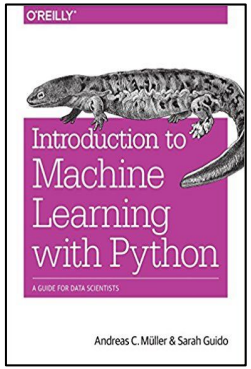
- En árboles: para un árbol, calcular importancia de permutación o importancia Gini.
- En ensambles: combinar importancias de cada árbol.
- En regresiones: Utilizar regularización (lasso, ridge, etc) y mirar los pesos.

Algoritmo: **RFE** (Recursive Feature Elimination)

1. Entrenar un modelo (árbol de decisión por ejemplo)
2. Obtener importancias a partir de un modelo.
3. Eliminar la / las variables menos importantes
4. Repetir

# LECTURA RECOMENDADA

Leer capítulo CHAPTER 4 Representing Data and Engineering Features del libro "Introduction to machine learning with Python: a guide for data scientists” (Müller, Andreas C., and Sarah Guido)



# Reducción dimensional

PCA

MDS

ISOMAP

T-SNE

# Reducción Dimensional y Manifold Learning

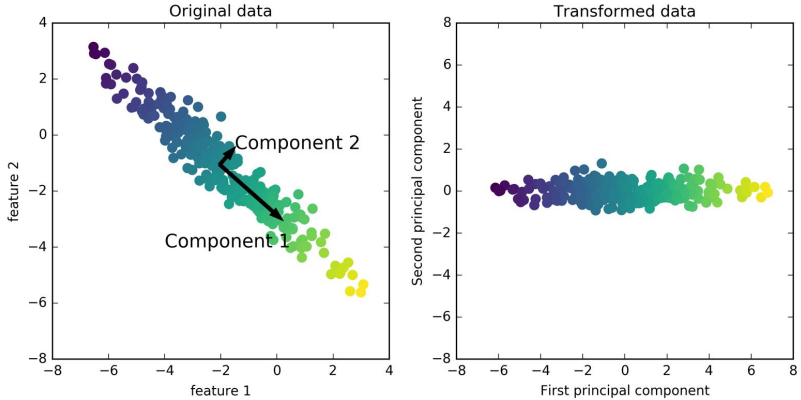
**Objetivos**

- Visualización.
- Reducción del ruido.
- Regularización de datos.
- Compresión de la información.
- Reducción del cómputo de los modelos.
- Etc.

# Reducción Dimensional y Manifold Learning

**Algunos algoritmos**

- **PCA (Principal Component Analysis)**
- MDS (Multidimensional Scaling)
- ISOMAP
- T-SNE (t-Stochastic Neighbor Embedding)




**PCA**

Idea: Rotar el dataset de manera de maximizar la varianza de los datos en proyecciones ortogonales.

**Suposición:**

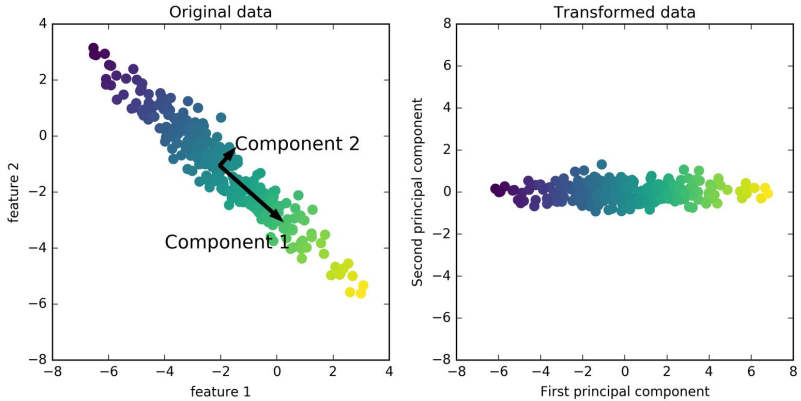
Los datos se encuentran mayormente en un subespacio lineal de dimensión menor a la original

[Bishop, Cap 12]

# Reducción Dimensional y Manifold Learning

## Algunos algoritmos

- **PCA (Principal Component Analysis)**
- MDS (Multidimensional Scaling)
- ISOMAP
- T-SNE (t-Stochastic Neighbor Embedding)




## PCA

Idea: Rotar el dataset de manera de maximizar la varianza de los datos en proyecciones ortogonales.

Es decir, encontrar:

$$\max_{\mathbf{u}}\{Var(\mathbf{u}^{\mathrm{T}}X)\} = \max_{\mathbf{u}}\{\mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u}\} \quad \text{con} \quad \mathbf{u}^{\mathrm{T}}\mathbf{u} = 1$$

Donde

$$\mathbf{S} = covar(\mathbf{X}) = \frac{1}{n-1}\sum_{i=1}^{n}(x^{(i)} - \overline{x})(x^{(i)} - \overline{x})^{\mathrm{T}}$$

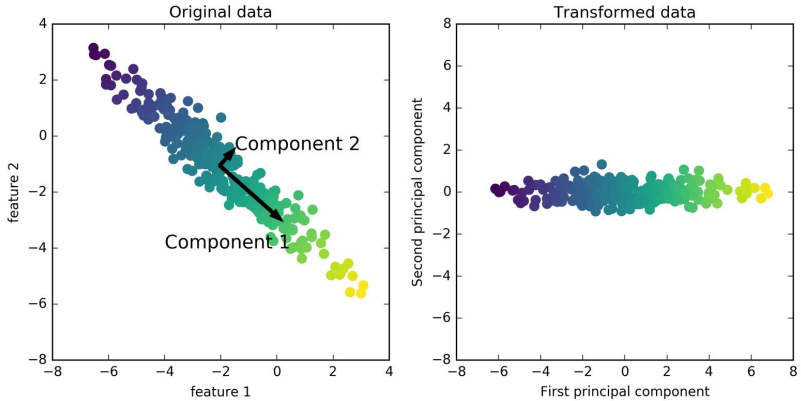
Obtenemos que las soluciones tienen la pinta

$\mathbf{S}\mathbf{u} = \lambda\mathbf{u}$ (autovectores de $\mathbf{S}$)

# Reducción Dimensional y Manifold Learning

## Algunos algoritmos

- **PCA (Principal Component Analysis)**
- MDS (Multidimensional Scaling)
- ISOMAP
- T-SNE (t-Stochastic Neighbor Embedding)




## PCA

Idea: Rotar el dataset de manera de maximizar la varianza de los datos en proyecciones ortogonales.

$$\max_{\mathbf{u}}\{Var(\mathbf{u}^{\mathrm{T}}X)\} = \max_{\mathbf{u}}\{\mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u}\} = \max_{\mathbf{u}}\{\mathbf{u}^{\mathrm{T}}\lambda\mathbf{u}\} = \max_{\mathbf{u}}\{\lambda\}$$

La dirección en la que se maximiza la varianza es la del **autovector** de **S** cuyo **autovalor** asociado es mayor.

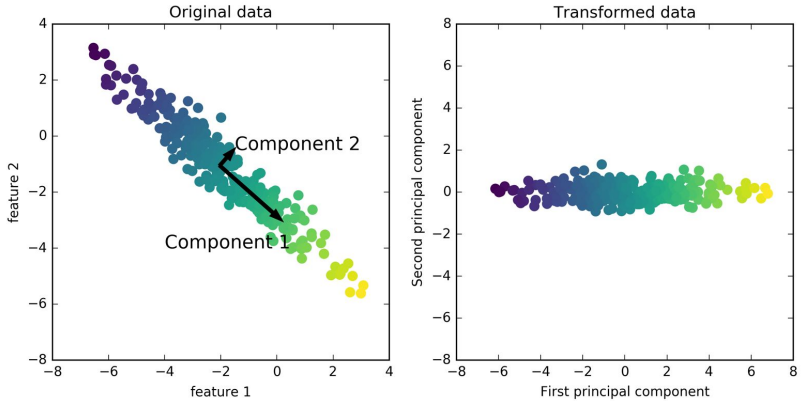
**u** será la **primera componente principal.**
**λ** será la **cantidad de varianza explicada** por **u.**

En la práctica, se implementa mediante aplicar **SVD** sobre una versión centrada de los datos.

# Reducción Dimensional y Manifold Learning

## Algunos algoritmos

- **PCA (Principal Component Analysis)**
- MDS (Multidimensional Scaling)
- ISOMAP
- T-SNE (t-Stochastic Neighbor Embedding)




## PCA

Idea: Rotar el dataset de manera de maximizar la varianza de los datos en proyecciones ortogonales.

Obtenidas las componentes principales, nos quedamos con las *N componentes* que mayor varianza expliquen.

Luego, podemos proyectar los datos y “reconstruir”.

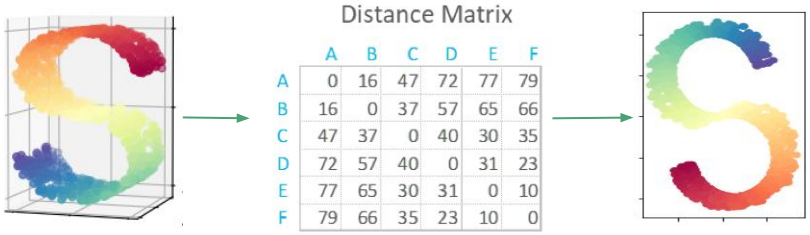
Proyectar: $\mathbf{z}^{(i)} = \mathbf{U}_d^T \mathbf{x}^{(i)}$

Reconstruir: $\hat{\mathbf{x}}^{(i)} = \mathbf{U}_d \mathbf{z}^{(i)}$

# Reducción Dimensional y Manifold Learning

**Algunos algoritmos**

- PCA (Principal Component Analysis)
- **MDS (Multidimensional Scaling)**
- ISOMAP
- T-SNE (t-Stochastic Neighbor Embedding)




**MDS**

Idea: Preservar las **distancias** entre puntos.

Ubicar los puntos en una dimensión menor tal que las distancias se parezcan lo más posible:

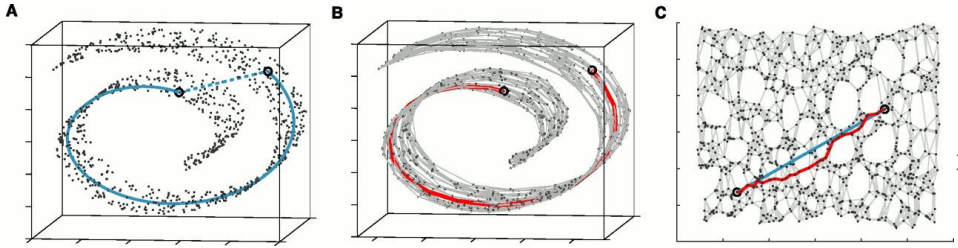
$$\min_{\mathbf{Z}} \{ \left( \sum_{i \neq j = 1,\ldots,n} \left( D(x^{(i)}, x^{(i)}) - D(z^{(i)} - z^{(j)}) \right)^2 \right\}$$

Si D = distancia euclidiana, el resultado es el de PCA. PCA obtiene el menor error de reconstrucción bajo esta distancia.

# Reducción Dimensional y Manifold Learning

**Algunos algoritmos**

- PCA (Principal Component Analysis)
- MDS (Multidimensional Scaling)
- **ISOMAP**
- T-SNE (t-Stochastic Neighbor Embedding)




**ISOMAP**

Idea: Preservar la **geometría** del manifold subyacente

MDS utilizando distancia geodésica. ¿Cómo calcularla?

1. Determinar vecinos más cercanos por cada punto
2. Construir un grafo de vecindad
   a. Se conectan todos los vecinos.
   b. Cada eje con longitud distancia euclidiana entre los puntos.
3. Computar el camino más cercano entre todo par de puntos:  Dijkstra's / Floyd–Warshall

# Reducción Dimensional y Manifold Learning

**Algunos algoritmos**

- PCA (Principal Component Analysis)
- MDS (Multidimensional Scaling)
- ISOMAP
- **T-SNE (t-Stochastic Neighbor Embedding)**

**Ver:** https://projector.tensorflow.org/

**Ver:** https://lvdmaaten.github.io/tsne/

**T-SNE**

Idea: Preservar la estructura **local** de los datos (no tanto la global)

1. Convertir las similitudes entre puntos en probabilidad de elegir un punto como vecino de otro.
2. Proyectar los puntos en la dimensión baja.
3. Convertir similitudes en probabilidades
4. Minimizar (mediante descenso de gradiente)

$$Costo = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log(\frac{p_{j|i}}{q_{j|i}})$$

# Resumen

- Conversión de variables
- Manejo de Missings
- Técnicas de selección de variables:
  - Filter
  - Embedded
  - Wrapped
- Reducción dimensional
  - PCA
  - MDS
  - ISOMAP
  - T-SNE